

Bi-Homomorphic Lattice-Based PRFs and Unidirectional Updatable Encryption^{*}

Vipin Singh Sehwat¹, Yvo Desmedt^{1,2}

¹ Department of Computer Science, The University of Texas at Dallas, USA

² Department of Computer Science, University College London, UK

Abstract. We define a pseudorandom function (PRF) $F : \mathcal{K} \times \mathcal{X} \rightarrow \mathcal{Y}$ to be bi-homomorphic when it is fully Key homomorphic and partially Input Homomorphic (KIH), i.e., given $F(k_1, x_1)$ and $F(k_2, x_2)$, there is an efficient algorithm to compute $F(k_1 \oplus k_2, x_1 \ominus x_2)$, where \oplus and \ominus are (binary) group operations. The homomorphism on the input is restricted to a fixed subset of the input bits, i.e., \ominus operates on some pre-decided m -out-of- n bits, where $|x_1| = |x_2| = n$, and the remaining $n - m$ bits are identical in both inputs. In addition, the output length, ℓ , of the operator \ominus is not fixed and is defined as $n \leq \ell \leq 2n$, hence leading to Homomorphically induced Variable input Length (HVL) as $n \leq |x_1 \ominus x_2| \leq 2n$. We present a learning with errors (LWE) based construction for a HVL-KIH-PRF family. Our construction is inspired by the key homomorphic PRF construction due to Banerjee and Peikert (Crypto 2014).

An updatable encryption scheme allows rotations of the encryption key, i.e., moving existing ciphertexts from old to new key. These updates are carried out via *update tokens*, which can be used by an untrusted party since the update procedure does not involve decryption of the ciphertext. We use our novel PRF family to construct an updatable encryption scheme, named QPC-UE-UU, which is quantum-safe, post-compromise secure and supports unidirectional ciphertext updates, i.e., the update tokens can be used to perform ciphertext updates but they cannot be used to undo already completed updates. Our PRF family also leads to the first left/right key homomorphic constrained-PRF family with HVL.

Keywords: Bi-Homomorphic PRF, Constrained-PRF, LWE, LWR, Updatable Encryption, Unidirectional Updates, Post-Compromise Security.

1 Introduction

In a PRF family [26], each function is specified by a short, random key, and can be easily computed given the key. Yet the function behaves like a random one, in the sense that if you are not given the key, and are

^{*} This is the full version of the paper that appears in Y. Mu et al. (Eds.): CANS 2019, LNCS 11829, pp. 1–21. DOI: 10.1007/978-3-030-31578-8_1 [53].

computationally bounded, then the input-output behavior of the function looks like that of a random function. Since their introduction, PRFs have been one of the most fundamental building blocks in cryptography. For a PRF F_s , the index s is called its key or seed. Many variants of PRFs with additional properties have been introduced and have found a plethora of applications in cryptography.

Key Homomorphic (KH) PRFs: A PRF family F is KH-PRF if the set of keys has a group structure and if there is an efficient algorithm that, given $F_s(x)$ and $F_t(x)$, outputs $F_{s+t}(x)$ [44]. Multiple KH-PRF constructions have been proposed via varying approaches [44, 13, 7, 46]. These functions have many applications in cryptography, such as, symmetric-key proxy re-encryption, updatable encryption, and securing PRFs against related-key attacks [13]. But, lack of input homomorphism limits the privacy preserving applications of KH-PRFs. For instance, while designing solutions for searchable symmetric encryption [21], it is highly desirable to hide the search patterns, which can be achieved by issuing random queries for each search. But, this feature cannot be supported if the search index is built by using a KH-PRF family, since it would require identical query (i.e., function input) in order to perform the same search.

Constrained PRFs (CPRFs): Constrained PRFs (also called delegatable PRFs) are another extension of PRFs. They enable a proxy to evaluate a PRF on a strict subset of its domain using a trapdoor derived from the CPRF secret key. A trapdoor is constructed with respect to a certain policy predicate that determines the subset of the input values for which the proxy is allowed to evaluate the PRF. Introduced independently by Kiayias et al. [35], Boneh et al. [14] and Boyle et al. [15] (termed functional PRFs), CPRFs have multiple interesting applications, including broadcast encryption, identify-based key exchange, batch query supporting searchable symmetric encryption and RFID. Banerjee et al. [6], and Brakerski and Vaikuntanathan [18] independently introduced KH-CPRFs.

Variable Input Length (VIL) PRFs: VIL-PRFs [9] serve an important role in constructing variable length block ciphers [10] and authentication codes [9], and are employed in prevalent protocols like Internet Key Exchange (IKEv2). No known CPRF or KH-CPRF construction supports variable input length.

Updatable Encryption (UE): In data storage, key rotation refers to the process of (periodically) exchanging the cryptographic key material that is used to protect the data. Key rotation is a desirable feature for cloud storage providers as it can be used to revoke old keys, that might have been comprised, or to enforce data access revocation. All major cloud storage providers (eg. Amazon’s Key Management Service [4], Google Cloud Platform [28]) implement some variants of data-at-rest encryption and hybrid encryption techniques to perform key rotation [23], which although efficient, do not support full key rotation as the procedures are designed to only update the key encapsulation but not the long-term key. Symmetric updatable encryption, introduced by Boneh et al. [13] (BLMR henceforth), supports full key rotation without performing decryption, i.e., the ciphertexts created under one key can be securely updated to ciphertexts generated via another key, with the help of a re-encryption/update token.

Everspaugh et al. [23] pointed out that the UE scheme from BLMR addresses relatively weak confidentiality goals and does not even consider integrity. They proposed a new security notion, named re-encryption indistinguishability, to better capture the idea of fully refreshing the keys upon rotation. They also presented an authenticated encryption based UE scheme, that satisfies the requirements of their new security model. Recently, Lehmann and Tackmann [38] observed that the previous security models/definitions (from BLMR and Everspaugh et al.) do not capture post-compromise security, i.e., the security guarantees after a key compromise. They also proved that neither of the two schemes is post-compromise secure under adaptive attacks. In the same paper, they presented the first UE scheme with post-compromise security. However, the security of that scheme is based on the Decisional Diffie Hellman (DDH) assumption, rendering it vulnerable to quantum computers [55]. It is important to note that all existing UE schemes only support bidirectional updates, i.e., the update token used to refresh the ciphertext for the current epoch can also be used to revert back to the previous epoch's ciphertext. Naturally, this is an undesirable feature. Hence, unidirectional updates [38], where the update tokens cannot be used to undo the ciphertext updates, is a highly desirable feature for UE schemes. But as mentioned earlier, no existing UE scheme achieves unidirectional updates.

1.1 Our Contributions

Our contributions can be classified into the following two broad classes.

1. **Novel PRF Classes and Constructions.** We introduce fully Key homomorphic and partially Input Homomorphic (KIH) PRFs with Homomorphically induced Variable input Length (HVL). A PRF, F , from such function family satisfies the condition that given $F_{k_1}(x_1)$ and $F_{k_2}(x_2)$, there exists an efficient algorithm to compute $F_{k_1 \oplus k_2}(x_1 \oplus x_2)$, where $|x_1 \oplus x_2| \geq |x_1|$ ($= |x_2|$) and the input homomorphism effects only some fixed m -out-of- n bits. We present a Learning with Errors (LWE) [51] based construction for such a PRF family. Our construction is inspired by the KH-PRF construction from Banerjee and Peikert [7]. A restricted case of our PRF family leads to another novel PRF class, namely left/right KH-CPRF with HVL.
2. **Quantum-Safe Post-Compromise Secure UE Scheme with Unidirectional Updates.** We use our HVL-KIH-PRF family to construct the first quantum-safe, post-compromise secure updatable encryption scheme with unidirectional updates. We know that the KH-PRF based UE scheme from BLMR is not post-compromise secure because it never updates the nonce [38]. Since our HVL-KIH-PRF family supports input homomorphism, in addition to key homomorphism, it allows updates to the nonce (i.e., the PRF input). Hence, we turn the BLMR UE scheme post-compromise secure by replacing their KH-PRF by our HVL-KIH-PRF, and introducing randomly sampled nonces for the ciphertext updates. The bi-homomorphism of our PRF family also allows us to enforce unidirectional updates. Since our PRF construction is based on the learning with errors (LWE) problem [51], our UE scheme is also quantum-safe.

1.2 Organization

Section 2 recalls the necessary background for the rest of the paper. Section 3 introduces and defines KIH-PRF, HVL-KIH-PRF and HVL-KH-CPRF. In Section 4, we present a LWE-based construction for a HVL-KIH-PRF family, provide its proof of correctness and discuss the different types of input homomorphisms that it supports. Section 5 gives the security proof for our HVL-KIH-PRF construction, while Section 6 analyzes its time complexity. Section 7 presents the construction of left/right HVL-KH-CPRF, that follows from a restricted case of our HVL-KIH-PRF family. In Section 8, we use of our HVL-KIH-PRF family to construct the first quantum-safe, post-compromise secure updatable encryption scheme with unidirectional updates. Section 9 discusses an interesting open problem, solving which would lead to a novel, search pattern hiding searchable encryption scheme. Section 10 gives the conclusion.

2 Background

This section recalls the necessary definitions required for the rest of the paper.

Definition 1 (Negligible Function). For security parameter ω , a function $\epsilon(\omega)$ is called *negligible* if for all $c > 0$ there exists a ω_0 such that $\epsilon(\omega) < 1/\omega^c$ for all $\omega > \omega_0$.

2.1 Learning with Errors

The learning with errors (LWE) problem requires to recover a secret s given a sequence of ‘approximate’ random linear equations on it. LWE is known to be hard based on certain assumptions regarding the worst-case hardness of standard lattice problems such as GapSVP (decision version of the Shortest Vector Problem) and SIVP (Shortest Independent Vectors Problem) [51, 47]. Many cryptosystems have been constructed whose security can be proven under the LWE problem, including (identity-based, leakage-resilient, fully homomorphic, functional) encryption [51, 24, 2, 42, 1, 17, 27], oblivious transfer [49], (blind) signatures [24, 40, 52, 41], PRFs [8], KH-PRFs [13, 7], KH-CPRFs [6, 18], hash functions [34, 48], etc.

Definition 2 (Decision-LWE [51]). For positive integers n and $q \geq 2$, and an error (probability) distribution $\chi = \chi(n)$ over \mathbb{Z}_q , the decision-LWE $_{n,q,\chi}$ problem is to distinguish between the following pairs of distributions:

$$(\mathbf{A}, \mathbf{A}^T \mathbf{s} + \mathbf{e}) \quad \text{and} \quad (\mathbf{A}, \mathbf{u}),$$

where $m = \text{poly}(n)$, $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \xleftarrow{\$} \chi^m$, and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$.

Definition 3 (Search-LWE [51]). For positive integers n and $q \geq 2$, and an error (probability) distribution $\chi = \chi(n)$ over \mathbb{Z}_q , the search-LWE $_{n,q,\chi}$ problem is to recover $\mathbf{s} \in \mathbb{Z}_q^n$, given $m(= \text{poly}(n))$ independent samples of $(\mathbf{A}, \mathbf{A}^T \mathbf{s} + \mathbf{e})$, where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, and $\mathbf{e} \xleftarrow{\$} \chi^m$.

Regev [51] showed that for a certain noise distribution χ and a sufficiently large q , the LWE problem is as hard as the worst-case SIVP (Shortest Independent Vectors Problem) and GapSVP (decision version of the Shortest Vector Problem) under a quantum reduction (see also [47, 16]). These results have been extended to show that \mathbf{s} can be sampled from a low norm distribution (in particular, from the noise distribution χ) and the resulting problem is as hard as the basic LWE problem [5]. Similarly, the noise distribution χ can be a simple low-norm distribution [43]. Note that the seed and error vectors in the definitions can be replaced by matrices of appropriate dimensions, that are sampled from the same distributions as the vectors. Such interchange does not affect the hardness of LWE [50].

2.2 Learning with Rounding

Based on a conjectured hard-to-learn function, Naor and Reingold [45] proposed synthesizers as a foundation to construct PRFs. At first glance, using LWE as the hard learning problem looks a valid option but Naor and Reingold’s synthesizer requires a deterministic hard-to-learn function, whereas LWE’s hardness depends the random, independent errors that are deliberately added to every output. In fact, without any error, LWE becomes trivially easy to learn. So, the main obstacle in constructing efficient lattice/LWE-based PRFs was finding a way to introduce (sufficiently independent) error terms into each of the exponentially many function outputs, while still keeping the function deterministic.

Banerjee et al. [8] introduced the LWR problem, in which instead of adding a small random error as done in LWE, a *deterministically* rounded version of the sample is released. In particular, for some $p < q$, the elements of \mathbb{Z}_q are divided into p contiguous intervals of roughly q/p elements each. The rounding function is defined as: $\lfloor \cdot \rfloor_p : \mathbb{Z}_q \rightarrow \mathbb{Z}_p$, that maps $x \in \mathbb{Z}_q$ into the index of the interval that x belongs to. Note that the error is introduced only when $q > p$, with “absolute” error being roughly equal to q/p , resulting in the “error rate” (relative to q) to be on the order of $1/p$. For a security parameter λ , they defined the rounding function $\lfloor \cdot \rfloor : \mathbb{Z}_q \rightarrow \mathbb{Z}_p$, where $q \geq p \geq 2$, as:

$$\lfloor x \rfloor_p = \left\lfloor \frac{p}{q} \cdot x \right\rfloor.$$

That is, if $\lfloor x \rfloor_p = i$, then $i \cdot \lfloor q/p \rfloor$ is the integer multiple of $\lfloor q/p \rfloor$ that is nearest to x . So, x is deterministically rounded to the nearest element of a sufficiently “coarse” public subset of $p \ll q$, well-separated values in \mathbb{Z}_q (e.g., a subgroup). Thus, the “error term” comes solely from deterministically rounding x to a relatively nearby value in \mathbb{Z}_p . The problem of distinguishing such rounded products from uniform samples is called the decision-learning with rounding problem, abbreviated as decision-LWR $_{n,q,p}$. Banerjee et al. proved decision-LWR $_{n,q,p}$ to be as hard as decision-LWE for a setting of parameters where the modulus and modulus-to-error ratio are super-polynomial.

Definition 4. Let $n \geq 1$ be the security parameter and moduli $q \geq p \geq 2$ be integers.

- For a vector $\mathbf{s} \in \mathbb{Z}_q^n$, define the LWR distribution $L_{\mathbf{s}}$ to be the distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_p$ obtained by choosing a vector $\mathbf{a} \leftarrow \mathbb{Z}_q^n$ uniformly at random, and outputting $(\mathbf{a}, b = \lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rfloor_p)$.

For a given distribution over $\mathbf{s} \in \mathbb{Z}_q^n$ (e.g., the uniform distribution), the decision-LWR $_{n,q,p}$ problem is to distinguish (with advantage non-negligible in n) between any desired number of independent samples $(\mathbf{a}_i, b_i) \leftarrow L_{\mathbf{s}}$, and the same number of samples drawn uniformly and independently from $\mathbb{Z}_q^n \times \mathbb{Z}_p$.

Theorem 1 ([8]). *Let χ be any efficiently sampleable B -bounded distribution over \mathbb{Z} , and let $q \geq p \cdot B \cdot \lambda(1)$, with λ being the security parameter. Then for any distribution over the secret $\mathbf{s} \in \mathbb{Z}_q^n$, solving the decision-LWR $_{n,q,p}$ problem is at least as hard as solving decision-LWE $_{n,q,\chi}$ for the same distribution over \mathbf{s} .*

Alwen et al. [3] gave a new reduction that works for a larger range of parameters, allowing for a polynomial modulus and modulus-to-error ratio. Bogdanov et al. [12] gave a more relaxed and general version of the theorem proved in [3]. LWR has been used to construct efficient pseudorandom generators and functions [8, 7]. Prior to the introduction of LWR, all constructions of weaker primitives such as symmetric authentication protocols [29, 30, 33], randomized weak PRFs [5], and message-authentication codes [36, 50] from noisy-learning problems were inherently randomized functions, where security relies on introducing fresh noise at every invocation. Due to its ease of use and efficiency, several schemes, such as Saber [22] and Round5 [11], along with some homomorphic encryption solutions [20], have based their hardness on LWR.

2.3 LWE-Based KH-PRFs

Due to small error being involved, LWE based KH-PRF constructions [13, 7] only achieve ‘almost homomorphism’, which is defined as:

Definition 5 ([13]). Let $F : \mathcal{K} \times \mathcal{X} \rightarrow \mathbb{Z}_p^m$ be an efficiently computable function such that (\mathcal{K}, \oplus) is a group. We say that the tuple (F, \oplus) is a γ -almost key homomorphic PRF if the following two properties hold:

1. F is a secure pseudorandom function.
2. For every $k_1, k_2 \in \mathcal{K}$ and every $x \in \mathcal{X}$, there exists a vector $\mathbf{e} \in [0, \gamma]^m$ such that: $F_{k_1}(x) + F_{k_2}(x) = F_{k_1 \oplus k_2}(x) + \mathbf{e} \bmod p$.

Boneh et al. [13] gave the first standard-model constructions of KH-PRFs using lattices/LWE. They proved their PRF to be secure under the m -dimensional (over an m -dimensional lattice) LWE assumption, for error rates $\alpha = m^{-\Omega(l)}$. Later, Banerjee and Peikert [7] gave KH PRFs from substantially weaker LWE assumptions, e.g., error rates $\alpha = m^{-\Omega(\log l)}$, yielding improved performance.

3 Novel PRF Classes: Definitions

In this section, we formally define KIH-PRF, HVL-KIH-PRF and HVL-KH-CPRF. For convenience, our definitions assume seed and error matrices instead of vectors. Note that such interchange does not affect the

hardness of LWE [50].

Notations. We begin by defining the important notations.

1. $x = x_\ell || x_r$, where $1 \leq |x_\ell| \leq \lfloor |x|/2 \rfloor$ and $|x_r| = |x| - |x_\ell|$.
2. $x_a = x_{a,\ell} || x_{a,r}$, where $1 \leq |x_{a,\ell}| \leq \lfloor |x_a|/2 \rfloor$ and $|x_{a,r}| = |x_a| - |x_{a,\ell}|$.

Assumption. Since the new PRF classes exhibit partial input homomorphism, without loss the generality, we assume x_r to be the homomorphic portion of the input x , with x_ℓ being the static/fixed portion. Obviously, the definitions remain valid if these are swapped, i.e., if x_ℓ is taken to be the homomorphic portion of the input with fixed/static x_r .

3.1 KIH-PRF

Definition 6. Let $F : \mathcal{K} \times \mathcal{X}' \times \mathcal{X} \rightarrow \mathbb{Z}_p^{m \times m}$ be a PRF family, such that (\mathcal{K}, \oplus) and (\mathcal{X}, \ominus) are groups. We say that the tuple (F, \ominus, \oplus) is a γ -almost fully key and partially input homomorphic PRF if the following condition holds:

- For every $k_1, k_2 \in \mathcal{K}$, with $x_{1,r}, x_{2,r} \in \mathcal{X}$ and $x_{1,\ell}, x_{2,\ell} \in \mathcal{X}'$, such that $x_{1,\ell} = x_{2,\ell} = x_\ell$, there exists a vector $\mathbf{E} \in [0, \gamma]^{m \times m}$ such that:

$$F_{k_1}(x_{1,\ell} || x_{1,r}) + F_{k_2}(x_{2,\ell} || x_{2,r}) + \mathbf{E} = F_{k_1 \oplus k_2}(x_\ell || x_r) \bmod p,$$

where $x_r = x_{1,r} \ominus x_{2,r}$.

3.2 HVL-KIH-PRF

Definition 7. Let $\mathcal{X} \subset \mathcal{Y}$, with \ominus defining the surjective mapping: $\mathcal{X} \ominus \mathcal{X} \rightarrow \mathcal{Y}$. Let $F : \mathcal{K} \times \mathcal{X}' \times \mathcal{X} \rightarrow \mathbb{Z}_p^{m \times m}$ and $F' : \mathcal{K} \times \mathcal{X}' \times \mathcal{Y} \rightarrow \mathbb{Z}_p^{m \times m}$ be two PRF families, where (\mathcal{K}, \oplus) is a group. We say that the tuple (F, \ominus, \oplus) is a γ -almost fully key and partially input homomorphic PRF with homomorphically induced variable input length, if the following condition holds:

- For every $k_1, k_2 \in \mathcal{K}$, and with $x_{1,r}, x_{2,r} \in \mathcal{X}$ and $x_{1,\ell}, x_{2,\ell} \in \mathcal{X}'$, such that $x_{1,\ell} = x_{2,\ell} = x_\ell$, there exists a vector $\mathbf{E} \in [0, \gamma]^{m \times m}$ such that:

$$F_{k_1}(x_{1,\ell} || x_{1,r}) + F_{k_2}(x_{2,\ell} || x_{2,r}) + \mathbf{E} = F'_{k_1 \oplus k_2}(x_\ell, y) \bmod p,$$

where $y = x_{1,r} \ominus x_{2,r}$.

3.3 Left/Right KH-CPRF with HVL

Definition 8. (Left KH-CPRF with HVL:) Let $\mathcal{X} \subset \mathcal{Y}$, with \ominus defining the surjective mapping: $\mathcal{X} \ominus \mathcal{X} \rightarrow \mathcal{Y}$. Let $F : \mathcal{K} \times \mathcal{W} \times \mathcal{X} \rightarrow \mathbb{Z}_p^{m \times m}$ and $F' : \mathcal{K} \times \mathcal{W} \times \mathcal{Y} \rightarrow \mathbb{Z}_p^{m \times m}$ be two PRF families, where (\mathcal{K}, \oplus) is a group.

We say that the tuple (F, \ominus, \oplus) is a left key homomorphic constrained-PRF with homomorphically induced variable input length, if for any $k_0 \in \mathcal{K}$ and a fixed $w \in \mathcal{W}$, given $F_{k_0}(w||x) \in \mathcal{F}$, where $x \in \mathcal{X}$, there exists an efficient algorithm to compute $F'_{k_0 \oplus k_1}(w, y) \in \mathcal{F}'$, for all $k_1 \in \mathcal{K}$ and $y \in \mathcal{Y}$.

(Right KH-CPRF with HVL:) Let $\mathcal{X} \subset \mathcal{Y}$, with \ominus defining the surjective mapping: $\mathcal{X} \ominus \mathcal{X} \rightarrow \mathcal{Y}$. Let $F : \mathcal{K} \times \mathcal{X} \times \mathcal{W} \rightarrow \mathbb{Z}_p^{m \times m}$ and $F' : \mathcal{K} \times \mathcal{Y} \times \mathcal{W} \rightarrow \mathbb{Z}_p^{m \times m}$ be two PRF families, where (\mathcal{K}, \oplus) is a group. We say that the tuple (F, \ominus, \oplus) is a right key homomorphic constrained-PRF with homomorphically induced variable input length, if for any $k_0 \in \mathcal{K}$ and a fixed $w \in \mathcal{W}$, given $F_{k_0}(x||w) \in \mathcal{F}$, where $x \in \mathcal{X}$, there exists an efficient algorithm to compute $F'_{k_0 \oplus k_1}(y, w) \in \mathcal{F}'$, for all $k_1 \in \mathcal{K}$ and $y \in \mathcal{Y}$.

4 LWE-Based HVL-KIH-PRF Construction

In this section, we present the first construction for a HVL-KIH-PRF family. Our construction is based on the LWE problem, and is inspired from the KH-PRF construction by Banerjee and Peikert [7].

4.1 Rounding Function

Let λ be the security parameter. Define a rounding function, $\lfloor \cdot \rfloor : \mathbb{Z}_q \rightarrow \mathbb{Z}_p$, where $q \geq p \geq 2$, as:

$$\lfloor x \rfloor_p = \left\lfloor \frac{p}{q} \cdot x \right\rfloor.$$

That is, if $\lfloor x \rfloor_p = i$, then $i \cdot \lfloor q/p \rfloor$ is the integer multiple of $\lfloor q/p \rfloor$ that is nearest to x . So, x is deterministically rounded to the nearest element of a sufficiently “coarse” public subset of $p \ll q$, well-separated values in \mathbb{Z}_q (e.g., a subgroup). Thus, the “error term” comes solely from deterministically rounding x to a relatively nearby value in \mathbb{Z}_p . As described in Section 2.2, the problem of distinguishing such rounded products from uniform samples is called the decision-learning with rounding (LWR _{n, q, p}) problem. The rounding function is extended component wise to vectors and matrices over \mathbb{Z}_q .

4.2 Definitions

Let $l = \lceil \log q \rceil$ and $d = l + 1$. Define a gadget vector as:

$$\mathbf{g} = (0, 1, 2, 4, \dots, 2^{l-1}) \in \mathbb{Z}_q^d.$$

Define a deterministic decomposition function $\mathbf{g}^{-1} : \mathbb{Z}_q \rightarrow \{0, 1\}^d$, such that $\mathbf{g}^{-1}(a)$ is a “short” vector and $\forall a \in \mathbb{Z}_q$, it holds that: $\langle \mathbf{g}, \mathbf{g}^{-1}(a) \rangle = a$, where $\langle \cdot \rangle$ denotes the inner product. The function \mathbf{g}^{-1} is defined as:

$$\mathbf{g}^{-1}(a) = (x', x_0, x_1, \dots, x_{l-1}) \in \{0, 1\}^d,$$

where $x' = 0$, and $a = \sum_{i=0}^{l-1} x_i 2^i$ is the binary representation of a . The gadget vector is used to define the gadget matrix \mathbf{G} as:

$$\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g} = \text{diag}(\mathbf{g}, \dots, \mathbf{g}) \in \mathbb{Z}_q^{n \times nd},$$

where \mathbf{I}_n is the $n \times n$ identity matrix and \otimes denotes the Kronecker product. The binary decomposition function, \mathbf{g}^{-1} , is applied entry-wise to vectors and matrices over \mathbb{Z}_q . Thus, \mathbf{g}^{-1} is extended to get another deterministic decomposition function $\mathbf{G}^{-1} : \mathbb{Z}_q^{n \times m} \rightarrow \{0, 1\}^{nd \times m}$, such that, $\mathbf{G} \cdot \mathbf{G}^{-1}(\mathbf{A}) = \mathbf{A}$. The addition operations *inside* the binary decomposition functions \mathbf{g}^{-1} and \mathbf{G}^{-1} are performed as simple integer operations (over all integers \mathbb{Z}), and not done in \mathbb{Z}_q .

4.3 Main Construction

The following notations are frequently used throughout this section.

- $x_{\ell h}$: left half of x , such that $|x_{\ell h}| = \lfloor |x|/2 \rfloor$,
- x_{rh} : right half of x , such that $|x_{rh}| = \lceil |x|/2 \rceil$,
- $x[i]$: i^{th} bit of bit-string x .

Let T be a full binary tree with at least one node, with $T.r$ and $T.\ell$ denoting its right and left subtree, respectively. For random matrices $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times nd}$, define function $\mathbf{A}_T : \{0, 1\}^{|T|} \rightarrow \mathbb{Z}_q^{n \times nd}$ recursively as:

$$\mathbf{A}_T(x) = \begin{cases} \mathbf{A}_x & \text{if } |T| = 1 \\ \mathbf{A}_{T.\ell}(x_\ell) + \mathbf{A}_{x[0]} \mathbf{G}^{-1}(\mathbf{A}_{T.r}(x_r)) & \text{otherwise,} \end{cases}$$

where $x = x_\ell || x_r$, $x_\ell \in \{0, 1\}^{|T.\ell|}$, $x_r \in \{0, 1\}^{|T.r|}$, and $|T|$ denotes the number of leaves in T . Based on the random seed $\mathbf{S} \in \mathbb{Z}_q^{n \times nd}$, the KIH-PRF family, $\mathcal{F}_{(\mathbf{A}_0, \mathbf{A}_1, T, p)}$, is defined as:

$$\mathcal{F}_{(\mathbf{A}_0, \mathbf{A}_1, T, p)} = \left\{ F_{\mathbf{S}} : \{0, 1\}^{2|T|} \longrightarrow \mathbb{Z}_p^{nd \times nd} \right\}.$$

Two *seed dependent* matrices, $\mathbf{B}_0, \mathbf{B}_1 \in \mathbb{Z}_q^{n \times nd}$, are defined as:

$$\mathbf{B}_0 = \mathbf{A}_0 + \mathbf{S}; \quad \mathbf{B}_1 = \mathbf{A}_1 + \mathbf{S},$$

Using the seed dependent matrices, a function $\mathbf{B}_T^{\mathbf{S}}(x)$ is defined recursively as:

$$\mathbf{B}_T^{\mathbf{S}}(x) = \begin{cases} \mathbf{B}_x & \text{if } |T| = 1 \\ \mathbf{B}_{T.\ell}^{\mathbf{S}}(x_\ell) + \mathbf{A}_{x[0]} \mathbf{G}^{-1}(\mathbf{B}_{T.r}^{\mathbf{S}}(x_r)) & \text{otherwise,} \end{cases}$$

Let $R : \{0, 1\}^{|T|} \rightarrow \mathbb{Z}_q^{nd \times n}$ be a pseudorandom generator. Let $y = y_{\ell h} || y_{rh}$, where $y_{\ell h}, y_{rh} \in \{0, 1\}^{|T|}$. In order to keep the length the equations in check, we represent the product $R(y_{\ell h}) \cdot \mathbf{A}_{y[0]}$ by the *notation*: $R_0(y_{\ell h})$. A member of the KIH-PRF family is indexed by the seed \mathbf{S} as:

$$F_{\mathbf{S}}(y) := [\mathbf{S}^T \cdot \mathbf{A}_T(y_{\ell h}) + R_0(y_{\ell h}) \cdot \mathbf{G}^{-1}(\mathbf{B}_T^{\mathbf{S}}(y_{rh}))]_p. \quad (1)$$

Let $\bar{0} = 00$, i.e., it represents two consecutive 0 bits. We define the following function family:

$$\mathcal{F}'_{(\mathbf{A}, T, p)} = \left\{ F'_{\mathbf{S}} : \{0, 1\}^{|T|} \times \{0, 1, \bar{0}\}^{|T|} \longrightarrow \mathbb{Z}_p^{nd \times nd} \right\},$$

where $\mathbb{A} = \{\mathbf{A}_0, \mathbf{A}_1, \mathbf{B}_0, \mathbf{B}_1, \mathbf{C}_0, \mathbf{C}_1, \overline{\mathbf{C}}_0\}$, and the matrices $\mathbf{C}_1, \mathbf{C}_0, \overline{\mathbf{C}}_0$ are defined by the seed $\mathbf{S} \in \mathbb{Z}_q^{n \times nd}$ as:

$$\mathbf{C}_1 = \mathbf{A}_0 + \mathbf{B}_1; \quad \overline{\mathbf{C}}_0 = \mathbf{A}_0 + \mathbf{B}_0; \quad \mathbf{C}_0 = \mathbf{A}_1 + \mathbf{B}_1.$$

Define a function $\mathbf{C}_T : \{0, 1\}^{|T|} \times \{0, 1, \bar{0}\}^{|T|} \rightarrow \mathbb{Z}_q^{n \times nd}$ recursively as:

$$\mathbf{C}_T^{\mathbf{S}}(x) = \begin{cases} \mathbf{C}_x & \text{if } |T| = 1 \\ \overline{\mathbf{C}}_0 & \text{if } |T| > 1 \wedge x[i] = x[i+1] = 0 \\ \mathbf{C}_{T,\ell}^{\mathbf{S}}(x_\ell) + \mathbf{A}_{x[0]} \mathbf{G}^{-1}(\mathbf{C}_{T,r}^{\mathbf{S}}(x_r)) & \text{otherwise,} \end{cases}$$

i.e., $\overline{\mathbf{C}}_0$ denotes two bits. Hence, during the evaluation of $\mathbf{C}_T^{\mathbf{S}}(x)$, a leaf in T may represent one bit or two bits. Let $z = z_0 || z_1$, where $z_0 \in \{0, 1\}^{|T|}$ and $z_1 \in \{0, 1, \bar{0}\}^{|T|}$. A member of the function family $\mathcal{F}'_{(\mathbb{A}, T, p)}$ is defined as:

$$F'_{\mathbf{S}}(z_0, z_1) := \lfloor \mathbf{S}^T \cdot \mathbf{A}_T(z_0) + R_0(z_0) \cdot \mathbf{G}^{-1}(\mathbf{C}_T^{\mathbf{S}}(z_1)) \rfloor_p, \quad (2)$$

where $R_0(z_0) = R(z_0) \cdot \mathbf{A}_{z_0[0]}$. Similar to the KH-PRF construction from [7], bulk of the computation performed while evaluating the PRFs, $F_{\mathbf{S}}(x)$ and $F'_{\mathbf{S}}(x_0, x_1)$, is in computing the functions $\mathbf{A}_T(x)$, $\mathbf{B}_T^{\mathbf{S}}(x)$, $\mathbf{C}_T^{\mathbf{S}}(x)$. While computing these functions on an input x , if all the intermediate matrices are saved, then $\mathbf{A}_T(x')$, $\mathbf{B}_T^{\mathbf{S}}(x')$, $\mathbf{C}_T^{\mathbf{S}}(x')$ can be incrementally computed for a x' that differs from x in a single bit. Specifically, one only needs to recompute the matrices for those internal nodes of T which appear on the path from the leaf representing the changed bit to the root. Hence, saving the intermediate matrices, that are generated while evaluating the functions on an input x can speed up successive evaluations on the related inputs x' .

4.4 Proof of Correctness

The homomorphically induced variable length (HVL) for our function family follows from the fact that $\{0, 1\}^{|T|} \subset \{0, 1, \bar{0}\}^{|T|}$. So, we move on to defining and proving the fully key and partially input homomorphic property for our function family. We begin by introducing a commutative binary operation, called ‘almost XOR’, which is denoted by $\bar{\oplus}$ and defined by the truth table given in Table 1.

$1 \bar{\oplus} 1$	$= 0$
$0 \bar{\oplus} 0$	$= 00$
$0 \bar{\oplus} 1$	$= 1$

Table 1: Truth Table for ‘almost XOR’ operation, $\bar{\oplus}$

Theorem 2. *For any inputs $x, y \in \{0, 1\}^{|T|}$ and a full binary tree $|T|$ such that: $x_{\ell h} = y_{\ell h} = z_0$ and $x_{rh} \bar{\oplus} y_{rh} = z_1$, where $z_0 \in \{0, 1\}^{|T|}$, and $z_1 \in \{0, 1, \bar{0}\}^{|T|}$, the following holds:*

$$F'_{(\mathbf{S}_1 + \mathbf{S}_2)}(z_0, z_1) = F_{\mathbf{S}_1}(x) + F_{\mathbf{S}_2}(y) + \mathbf{E}, \quad (3)$$

where $\|\mathbf{E}\|_{\infty} \leq 1$.

Proof. We begin by making an important observation, and arguing about its correctness.

Observation 1 The HVL-KIH-PRF family, defined in Equation. 1, requires both addition and multiplication operations for each function evaluation. Hence, adding the outputs of two functions, $F_{\mathbf{S}_1}$ and $F_{\mathbf{S}_2}$, from the function family \mathcal{F} translates into adding the outputs per node of the tree T . As a result, the decomposition function \mathbf{G}^{-1} for each node in T takes one of the following three forms:

1. $\mathbf{G}^{-1}(\mathbf{A}_b + \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\cdot)),$
2. $\mathbf{G}^{-1}(\mathbf{A}_b + \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\cdot) + \cdots + \mathbf{A}_{x_0} \cdot \mathbf{G}^{-1}(\cdot)),$
3. $\mathbf{G}^{-1}(\mathbf{A}_b), \quad b \in \{0, 1\},$

where $\mathbf{G}^{-1}(\cdot)$ denotes possibly nested \mathbf{G}^{-1} . Note that each term $\mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\cdot)$ or a summation of such terms, i.e., $\mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\cdot) + \cdots + \mathbf{A}_{x_0} \cdot \mathbf{G}^{-1}(\cdot)$, yields some matrix $\check{\mathbf{A}} \in \mathbb{Z}_q^{n \times nd}$. Hence, each decomposition function \mathbf{G}^{-1} in the recursively unwound function $F'_{(\mathbf{S}_1 + \mathbf{S}_2)}(z_0, z_1)$ has at most two “direct” inputs/arguments (i.e., \mathbf{A}_b and $\check{\mathbf{A}}$).

Recall from Section 4.2 that for the “direct” arguments of \mathbf{G}^{-1} , addition operations are performed as simple integer operations (over all integers \mathbb{Z}) instead of being done in \mathbb{Z}_q . We know from Observation 1, that unwinding of the recursive function $F'_{\mathbf{S}}(z_0, z_1)$ yields each binary decomposition function \mathbf{G}^{-1} with at most two “direct” inputs. We also know that binary decomposition functions (\mathbf{g}^{-1} and \mathbf{G}^{-1}) are linear, provided there is no carry bit or there is an additional bit to accommodate the possible carry. Hence, by virtue of the extra bit, $d - l$ (where $l = \lceil \log q \rceil$, and $d = l + 1$), each \mathbf{G}^{-1} behaves as a linear function during the evaluation of our function families, i.e., the following holds:

$$\mathbf{G}^{-1}(\mathbf{A}_i + \mathbf{A}_j) = \mathbf{G}^{-1}(\mathbf{A}_i) + \mathbf{G}^{-1}(\mathbf{A}_j), \quad (4)$$

where $\mathbf{G}^{-1}(\mathbf{A}_i) + \mathbf{G}^{-1}(\mathbf{A}_j)$ is component-wise vector addition of the n, d bits long bit vectors of the columns, $[\mathbf{v}_1, \dots, \mathbf{v}_{nd}] \in \mathbb{Z}^{1 \times nd}$, of $\mathbf{G}^{-1}(\mathbf{A}_i)$ with the n, d bits long bit vectors of the columns, $[\mathbf{w}_1, \dots, \mathbf{w}_{nd}] \in \mathbb{Z}_q^{1 \times nd}$, of $\mathbf{G}^{-1}(\mathbf{A}_j)$. We are now ready to prove Equation 3. Since $y_{\ell h} = x_{\ell h}$, we use $x_{\ell h}$ to represent both, as that helps clarity. Let $\mathbf{S} = \mathbf{S}_1 + \mathbf{S}_2$, then by using Equation 2, we can write the LHS of Equation 3 as: $[\mathbf{S}^T \cdot \mathbf{A}_T(z_0) + R_0(z_0) \cdot \mathbf{G}^{-1}(\mathbf{C}_T^{\mathbf{S}}(z_1))]_p$.

Similarly, from Equation 1, we get RHS of Equation 3 equal to:

$$[\mathbf{S}_1^T \cdot \mathbf{A}_T(x_{\ell h}) + R_0(x_{\ell h}) \cdot \mathbf{G}^{-1}(\mathbf{B}_T^{\mathbf{S}_1}(x_{rh}))]_p + [\mathbf{S}_2^T \cdot \mathbf{A}_T(x_{\ell h}) + R_0(x_{\ell h}) \cdot \mathbf{G}^{-1}(\mathbf{B}_T^{\mathbf{S}_2}(y_{rh}))]_p + \mathbf{E}.$$

We know that $\lfloor a + b \rfloor_p = \lfloor a \rfloor_p + \lfloor b \rfloor_p + e$. We further know that $x_{\ell h} = z_0$, and $R_0(x_{\ell h}) = R_0(z_0)$. Thus, from Equation 4, the RHS can be written as:

$$\begin{aligned} & [(\mathbf{S}_1 + \mathbf{S}_2)^T \cdot \mathbf{A}_T(z_0) + R_0(z_0) \cdot \mathbf{G}^{-1}(\mathbf{B}_T^{\mathbf{S}_1}(x_{rh}) + \mathbf{B}_T^{\mathbf{S}_2}(y_{rh}))]_p \\ &= [\mathbf{S}^T \cdot \mathbf{A}_T(z_0) + R_0(z_0) \cdot \mathbf{G}^{-1}(\mathbf{C}_T^{\mathbf{S}}(x_{rh} \oplus y_{rh}))]_p \\ &= [\mathbf{S}^T \cdot \mathbf{A}_T(z_0) + R_0(z_0) \cdot \mathbf{G}^{-1}(\mathbf{C}_T^{\mathbf{S}}(z_1))]_p = LHS. \end{aligned}$$

□

5 Security Proof

The security proofs as well as the time complexity analysis of our construction depend on the tree T . Left and right depth of T are respectively defined as the maximum left and right depths over all leaves in T . The modulus q , the underlying LWE error rate, and the dimension n needed to obtain a desired level of provable security, are largely determined by two parameters of T . The first one, called *expansion* $e(T)$ [7], is defined as:

$$e(T) = \begin{cases} 0 & \text{if } |T| = 1 \\ \max\{e(T.\ell) + 1, e(T.r)\} & \text{otherwise.} \end{cases}$$

For our construction, $e(T)$ is the maximum number of terms of the form $\mathbf{G}^{-1}(\cdot)$ that get consecutively added together when we unwind the recursive definition of the function \mathbf{A}_T . The second parameter is called *sequentiality* [7], which gives the maximum number of nested \mathbf{G}^{-1} functions, and is defined as:

$$s(T) = \begin{cases} 0 & \text{if } |T| = 1 \\ \max\{s(T.\ell), s(T.r) + 1\} & \text{otherwise.} \end{cases}$$

For our function families, over the uniformly random and independent choice of $\mathbf{A}_0, \mathbf{A}_1, \mathbf{S} \in \mathbb{Z}_q^{n \times nd}$, and with the secret key chosen uniformly from \mathbb{Z}_q^n , the modulus-to-noise ratio for the underlying LWE problem is: $q/r \approx (n \log q)^{e(T)}$. Known reductions [51, 47, 16] (for $r \geq 3\sqrt{n}$) guarantee that such a LWE instantiation is at least as hard as approximating hard lattice problems like GapSVP and SIVP, in the worst case to within $\approx q/r$ factors on n -dimensional lattices. Known algorithms for achieving such factors take time exponential in $n/\log(q/r) = \tilde{\Omega}(n/e(T))$. Hence, in order to obtain provable 2^λ (where λ is the input length) security against the best known lattice algorithms, the best parameter values are the same as defined for the KH-PRF construction from [7], which are:

$$n = e(T) \cdot \tilde{\Theta}(\lambda) \quad \text{and} \quad \log q = e(T) \cdot \tilde{\Theta}(1). \quad (5)$$

5.1 Overview of KH-PRF from [7]

As mentioned earlier, our construction is inspired by the KH-PRF construction from [7]. Our security proofs rely on the security of that construction. Therefore, before moving to the security proofs, it is necessary that we briefly recall the KH-PRF construction from [7]. Although that scheme differs from our KIH PRF construction, certain parameters and their properties are identical. The rounding function $\lfloor \cdot \rfloor_p$, binary tree T , gadget vector/matrix \mathbf{g}/\mathbf{G} , the binary decomposition functions $\mathbf{g}^{-1}/\mathbf{G}^{-1}$ and the base matrices $\mathbf{D}_0, \mathbf{D}_1$ in that scheme are defined similarly to our construction. There is a difference in the definitions of the decomposition functions, which for our construction are defined as (see Section 4.2): $\mathbf{g}^{-1} : \mathbb{Z}_q \rightarrow \{0, 1\}^d$ and $\mathbf{G}^{-1} : \mathbb{Z}_q^{n \times m} \rightarrow \{0, 1\}^{nd \times m}$, i.e., the dimensions of the output space for our decomposition functions has $d(=l+1)$ instead of $l = \lceil \log q \rceil$ as in [7]. Recall that the extra (carry) bit ensures that Equation 4 holds.

KH-PRF Construction from [7] Given two uniformly selected matrices, $\mathbf{D}_0, \mathbf{D}_1 \in \mathbb{Z}_q^{n \times nl}$, define function $\mathbf{D}_T(x) : \{0, 1\}^{|T|} \rightarrow \mathbb{Z}_q^{n \times nl}$ as:

$$\mathbf{D}_T(x) = \begin{cases} \mathbf{D}_x & \text{if } |T| = 1 \\ \mathbf{D}_{T.\ell}(x_{\ell t}) \cdot \mathbf{G}^{-1}(\mathbf{D}_{T.r}(x_{rt})) & \text{otherwise,} \end{cases} \quad (6)$$

where $x = x_{\ell t} || x_{rt}$, for $x_{\ell t} \in \{0, 1\}^{|T.\ell|}$, $x_{rt} \in \{0, 1\}^{|T.r|}$. The KH-PRF function family is defined as:

$$\mathcal{H}_{\mathbf{D}_0, \mathbf{D}_1, T, p} = \left\{ H_{\mathbf{s}} : \{0, 1\}^{|T|} \rightarrow \mathbb{Z}_p^{nl} \right\},$$

where $p \leq q$ is the modulus. A member of the function family \mathcal{H} is indexed by the seed $\mathbf{s} \in \mathbb{Z}_q^n$ as: $H_{\mathbf{s}}(x) = \lfloor \mathbf{s} \cdot \mathbf{D}_T(x) \rfloor_p$.

For the sake of completeness, we recall the main security theorem from [7].

Theorem 3 ([7]). *Let T be any full binary tree, χ be some distribution over \mathbb{Z} that is subgaussian with parameter $r > 0$ (e.g., a bounded or discrete Gaussian distribution with expectation zero), and*

$$q \geq p \cdot r \sqrt{|T|} \cdot (nl)^{e(T)} \cdot \lambda^{\omega(1)},$$

where λ is the input size. Then over the uniformly random and independent choice of $\mathbf{D}_0, \mathbf{D}_1 \in \mathbb{Z}_q^{n \times nl}$, the family $\mathcal{H}_{\mathbf{D}_0, \mathbf{D}_1, T, p}$ with secret key chosen uniformly from \mathbb{Z}_q^n is a secure PRF family, under the decision-LWE $_{n, q, \chi}$ assumption.

5.2 Security Proof of Our Construction

The dimensions and bounds for the parameters r, q, p, n, m and χ in our construction are the same as in [7]. We begin by defining the necessary terminology.

1. Reverse-LWE: is an LWE instance $\mathbf{S}^T \mathbf{A} + \mathbf{E}$ with secret lattice-basis \mathbf{A} and public seed matrix \mathbf{S} .
2. Reverse-LWR: is defined similarly, i.e., $\lfloor \mathbf{S}^T \mathbf{A} \rfloor_p$ with secret \mathbf{A} and public \mathbf{S} .
3. If H represents the binary entropy function, then we know that for uniformly random $\mathbf{A} \in \mathbb{Z}_q^{n \times nd}$ and a random seed $\mathbf{S} \in \mathbb{Z}_q^{n \times nd}$, it holds that: $H(\mathbf{A}) = H(\mathbf{S})$. Hence, it follows from elementary linear algebra that reverse-LWR $_{n, q, p}$ and reverse-LWE $_{n, q, \chi}$ are at least as hard as decision-LWR $_{n, q, p}$ and decision-LWE $_{n, q, \chi}$, respectively.

Observation 2 Consider the function family $\mathcal{F}_{(\mathbf{A}_0, \mathbf{A}_1, T, p)}$. We know that a member of the function family is defined by a random seed $\mathbf{S} \in \mathbb{Z}_q^{n \times nd}$ as:

$$F_{\mathbf{S}}(x) = \lfloor \mathbf{S}^T \cdot \mathbf{A}_T(x_{\ell h}) + R_0(x_{\ell h}) \cdot \mathbf{G}^{-1}(\mathbf{B}_T^{\mathbf{S}}(x_{rh})) \rfloor_p = \underbrace{\lfloor \mathbf{S}^T \cdot \mathbf{A}_T(x_{\ell h}) \rfloor_p}_{\mathbf{L}_T(x_{\ell h})} + \underbrace{\lfloor R_0(x_{\ell h}) \cdot \mathbf{G}^{-1}(\mathbf{B}_T^{\mathbf{S}}(x_{rh})) \rfloor_p}_{\mathbf{R}_T(x_{rh})} + \mathbf{E}.$$

Observation 3 For $|T| \geq 1$ and $x \in \{0, 1\}^{2|T|}$, each $\mathbf{L}_T(x_{\ell h})$ is the sum of the following three types of terms:

1. Exactly one term of the form: $\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \rfloor_p$, corresponding to the leftmost child of the full binary tree T and the most significant bit, $x[0]$, of x .
2. At least one term of the following form:

$$\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{A}_{x[i]}) \rfloor_p; \quad (1 \leq i \leq 2|T|),$$

corresponding to the right child at level 1 of the full binary tree T .

3. Zero or more terms with nested \mathbf{G}^{-1} functions of the form:

$$\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{A}_{x[i]} + \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\cdot) + \dots) \rfloor_p; \quad (1 \leq i \leq 2|T|).$$

We prove that for appropriate parameters and input length, each one of the aforementioned terms is a pseudorandom function on its own.

Lemma 1. *Let n be a positive integer, $q \geq 2$ be the modulus, χ be a probability distribution over \mathbb{Z} , and m be polynomially bounded (i.e. $m = \text{poly}(n)$). For a uniformly random and independent choice of $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times nd}$ and a random seed vector $\mathbf{S} \in \mathbb{Z}_q^{n \times nd}$, the function family $\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[i]} \rfloor_p$ for the single bit input $x[i]$ ($1 \leq i \leq 2|T|$) is a secure PRF family under the decision-LWE $_{n,q,\chi}$ assumption.*

Proof. We know from [8] that $\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[i]} \rfloor_p$ is a secure PRF under the decision-LWR $_{n,q,p}$ assumption, which is at least as hard as solving the decision-LWE $_{n,q,\chi}$ problem. \square

Corollary 1 (To Theorem 3) *Let n be a positive integer, $q \geq 2$ be the modulus, χ be a probability distribution over \mathbb{Z} , and $m = \text{poly}(n)$. For uniformly random and independent matrices $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times nd}$ with a random seed $\mathbf{S} \in \mathbb{Z}_q^{n \times nd}$, the function $\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{A}_{x[i]}) \rfloor_p$ for the two bit input: $x[0]||x[i]$, is a secure PRF family, under the decision-LWE $_{n,q,\chi}$ assumption.*

Proof. For the two bit input $x[0]||x[i]$, the expression $\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{A}_{x[i]}) \rfloor_p$ is an instance of the function $\mathcal{H}_{\mathbf{A}_0, \mathbf{A}_1, T, p}$ (see Section 5.1). Hence, it follows from Theorem 3 that $\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{A}_{x[i]}) \rfloor_p$ is a secure PRF family under the decision-LWE $_{n,q,\chi}$ assumption. \square

Corollary 2 (To Theorem 3) *Let n be a positive integer, $q \geq 2$ be the modulus, χ be a probability distribution over \mathbb{Z} , and $m = \text{poly}(n)$. Given uniformly random and independent $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times nd}$, and a random seed $\mathbf{S} \in \mathbb{Z}_q^{n \times nd}$, the function: $\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{A}_{x[i]} + \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\cdot) + \dots) \rfloor_p$ is a secure PRF family under the decision-LWE $_{n,q,\chi}$ assumption.*

Proof. Since, $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times nd}$ are random and independent, $\mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\cdot)$ is statistically indistinguishable from $\mathbf{A}_{x[i]} \cdot \mathbf{G}^{-1}(\cdot)$, as defined by the function $\mathbf{B}_T(x)$ (see Equation 6), where $\mathbf{G}^{-1}(\cdot)$ represents possibly nested \mathbf{G}^{-1} . Hence, it follows from Theorem 3 that for the “right spine” (with leaves for all the left children) full binary tree T , $\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{A}_{x[i]} + \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\cdot) + \dots) \rfloor_p$ defines a secure PRF family under the decision-LWE $_{n,q,\chi}$ assumption. \square

Corollary 3 (To Theorem 3) *Let n be a positive integer, $q \geq 2$ be the modulus, χ be a probability distribution over \mathbb{Z} , and $m = \text{poly}(n)$. For uniformly random and independent matrices $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times nd}$, and a random seed $\mathbf{S} \in \mathbb{Z}_q^{n \times nd}$, the function $\mathbf{R}_T(x_{rh}) = \lfloor R_0(x_{\ell h}) \cdot \mathbf{G}^{-1}(\mathbf{B}_T^{\mathbf{S}}(x_{rh})) \rfloor_p$ is a secure PRF family under the decision-LWE $_{n,q,\chi}$ assumption.*

Proof. We know that $\mathbf{A}_0, \mathbf{A}_1, \mathbf{S} \in \mathbb{Z}_q^{n \times nd}$ are generated uniformly and independently. Therefore, the secret matrices, $\mathbf{B}_0, \mathbf{B}_1$, defined as: $\mathbf{B}_0 = \mathbf{A}_0 + \mathbf{S}$ and $\mathbf{B}_1 = \mathbf{A}_1 + \mathbf{S}$, have the same distribution as $\mathbf{A}_0, \mathbf{A}_1$. As $R : \{0, 1\}^{|T|} \rightarrow \mathbb{Z}_q^{nd \times n}$ is a PRG, $R(x_{\ell h})$ is a valid seed matrix for decision-LWE, making $\mathbf{R}_T(x_{rh})$ an instance of reverse-LWR $_{n,q,p}$, which we know is as hard as the decision-LWR $_{n,q,p}$ problem. Hence, it follows from Theorem 3 that $\mathbf{R}_T(x_{rh})$ defines a secure PRF family for secret $R(x_{\ell h})$. \square

Theorem 4. *Let T be any full binary tree, χ be some distribution over \mathbb{Z} that is subgaussian with parameter $r > 0$ (e.g., a bounded or discrete Gaussian distribution with expectation zero), $R : \{0, 1\}^{|T|} \rightarrow \mathbb{Z}_q^{nd \times n}$ be a PRG, and*

$$q \geq p \cdot r \sqrt{|T|} \cdot (nd)^{e(T)} \cdot \lambda^{\omega(1)}.$$

Then over the uniformly random and independent choice of $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times nd}$ and a random seed $\mathbf{S} \in \mathbb{Z}_q^{n \times nd}$, the family $\mathcal{F}_{(\mathbf{A}_0, \mathbf{A}_1, T, p)}$ is a secure PRF under the decision-LWE $_{n,q,\chi}$ assumption.

Proof. From Observations 2 and 3, we know that each member $F_{\mathbf{S}}$ of the function family $\mathcal{F}_{(\mathbf{A}_0, \mathbf{A}_1, T, p)}$ is defined by the random seed \mathbf{S} , and can be written as:

$$\begin{aligned} F_{\mathbf{S}}(x) = & \lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \rfloor_p + d_1 \cdot \lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{A}_{x[i]}) \rfloor_p + \\ & d_2 \cdot \lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{A}_{x[i]} + \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\cdot) + \dots) \rfloor_p + \\ & d_3 \cdot \lfloor R(x_{\ell h}) \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{B}_T^{\mathbf{S}}(x_{rh})) \rfloor_p + \mathbf{E}, \end{aligned}$$

where $d_1, d_2, d_3 \in \mathbb{Z}$, such that, $1 \leq d_1, d_3 \leq |T|$ and $0 \leq d_2 \leq |T|$. From Lemma 1, and Corollaries 1, 2 and 3, we know that the following are secure PRFs under the decision-LWE $_{n,q,\chi}$ assumption:

1. $\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \rfloor_p, \lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{A}_{x[i]}) \rfloor_p,$
2. $\lfloor \mathbf{S}^T \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{A}_{x[i]} + \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\cdot) + \dots) \rfloor_p,$
3. $\lfloor R(x_{\ell h}) \cdot \mathbf{A}_{x[0]} \cdot \mathbf{G}^{-1}(\mathbf{B}_T^{\mathbf{S}}(x_{rh})) \rfloor_p.$

Hence, it follows that the function family $\mathcal{F}_{(\mathbf{A}_0, \mathbf{A}_1, T, p)}$ is a secure PRF under the decision-LWE $_{n,q,\chi}$ assumption.

Corollary 4 (To Theorem 4) *Let T be any full binary tree, χ be some distribution over \mathbb{Z} that is subgaussian with parameter $r > 0$ (e.g., a bounded or discrete Gaussian distribution with expectation zero), $R : \{0, 1\}^{|T|} \rightarrow \mathbb{Z}_q^{nd \times n}$ be a PRG, and*

$$q \geq p \cdot r \sqrt{|T|} \cdot (nd)^{e(T)} \cdot \lambda^{\omega(1)}.$$

Then over the uniformly random and independent choice of $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times nd}$ and random seed $\mathbf{S} \in \mathbb{Z}_q^{n \times nd}$, the family $\mathcal{F}'_{(\mathbf{A}, T, p)}$ is a secure PRF under the decision-LWE $_{n,q,\chi}$ assumption.

6 Time Complexity Analysis

In this section, we analyze the asymptotic time complexity of evaluating a function from our HVL-KIH-PRF family. We know that the time complexity of the binary decomposition function \mathbf{g}^{-1} is $O(\log q)$, and that \mathbf{G}^{-1} is simply \mathbf{g}^{-1} applied entry-wise. The size of the public matrices, $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times nd}$, is $\Theta(n^2 \log q)$, which by Equation 5 is $e(T)^4 \cdot \tilde{\Theta}(\lambda^2)$ bits. The secret matrix, $\mathbf{S} \in \mathbb{Z}_q^{n \times nd}$, also has the same size, i.e., $e(T)^4 \cdot \tilde{\Theta}(\lambda^2)$. Computing $\mathbf{A}_T(x), \mathbf{B}_T(x)$ or $\mathbf{C}_T(x)$ requires one decomposition with \mathbf{G}^{-1} , one $(n \times nd)$ -by- $(nd \times nd)$ matrix multiplication and one $(n \times nd)$ -by- $(n \times nd)$ matrix addition over \mathbb{Z}_q , per internal node of T . Hence, the total time complexity comes out to be $\Omega(|T| \cdot n^\omega \log^2 q)$ operations in \mathbb{Z}_q , where $\omega \geq 2$ is the exponent of matrix multiplication.

7 Left/Right HVL-KH-CPRFs

In this section, we present the construction of another novel PRF class, namely left/right HVL-KH-CPRFs, as a special case of our HVL-KIH-PRF family. Let $F' : \mathcal{K} \times \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{Z}$ be the PRF defined by Equation 2. The goal is to derive a *constrained* key PRF, $k_{x, \text{left}}$ or $k_{x, \text{right}}$ for every $x \in \mathcal{X}$ and $k_0 \in \mathcal{K}$, such that $k_{x, \text{left}} = F_{k_0}(x || \cdot)$ (where $F : \mathcal{K} \times \mathcal{X} \times \mathcal{X} \rightarrow \mathcal{Z}$ is the PRF family defined by Equation 1) enables the evaluation of the PRF function $F'_k(x, y)$ for the key $k = k_0 + k_1$, where $k_1 \in \mathcal{K}$, and the subset of points $\{(x, y) : y \in \mathcal{Y}\}$, i.e., all the points where the left portion of the input is x . Similarly, the constrained key $k_{x, \text{right}} = F_{k_0}(\cdot || x)$ enables the evaluation of the PRF function $F'_k(x, y)$ for the key $k = k_0 + k_1$, where $k_1 \in \mathcal{K}$, and the subset of points $\{(y, x) : y \in \mathcal{Y}\}$, i.e., all the points where the right side of the input is x .

KH-CPRF Construction. We begin by giving a construction for left KH-CPRF, without HVL, and then turn it into a HVL-KH-CPRF construction. Our HVL-KIH-PRF function, defined in Equation 1, is itself a left KH-CPRF when evaluated as: $F_{k_0}(x_0 || \mathbf{1})$, i.e., the key is $k_0 \in \mathcal{K}$, the left side of the input is $x_0 \in \mathcal{X}$, and the right half is an all one vector, $\mathbf{1} = \{1\}^{\log |\mathcal{X}|}$. Now, to evaluate $F'_k(x_0, x_1)$ at a key $k = k_0 + k_1$, and any right input $x_1 \in \mathcal{Y}$, first evaluate $F_{k_1}(x_0 || x'_1)$ and add its output with that of the given constrained function, $F_{k_0}(x_0 || \mathbf{1})$, i.e., compute: $F'_k(x_0, x_1) = F_{k_1}(x_0 || x'_1) + F_{k_0}(x_0 || \mathbf{1})$, where $x'_1 \in \mathcal{X}$, and $x_1 = x'_1 \bar{\oplus} \mathbf{1}$, with $k = k_0 + k_1$. Recall from Table 1 that ‘almost XOR’, $\bar{\oplus}$, differs from XOR only for the case when both inputs are zero. Hence, having $\mathbf{1}$ as the right half effectively turns $\bar{\oplus}$ into \oplus , and ensures that all possible right halves $x_1 \in \mathcal{X}$ can be realized via $x'_1 \in \mathcal{X}$.

Similarly, right KH-CPRF can be realized by provisioning the constrained function $F_{k_0}(\mathbf{1} || x_0)$, where $\mathbf{1} = \{1\}^{\log |\mathcal{X}|}$ is an all ones vector and $x_0 \in \mathcal{X}$. This interchange allows one to evaluate a different version of our HVL-KIH-PRF function, where the left portion of the input exhibits homomorphism (see Section 3). Hence, for all $k_1 \in \mathcal{K}$ and $x_1 \in \mathcal{X}$, it supports evaluation of $F'_{k_0+k_1}(x_1 || x_0)$.

Achieving HVL. Wlog, we demonstrate left In order to achieve HVL for the left/right KH-CPRF given above, simply replace the all ones vector, $\mathbf{1}$, with an all zeros vector, $\mathbf{0}$, of the same dimension. Hence, the new constrained functions become: $k_{\text{left}} = F_{k_0}(x_0 || \mathbf{0})$ and $k_{\text{right}} = F_{k_0}(\mathbf{0} || x_0)$. This enables us to evaluate

$F'_k(x_0, x_1) \in \mathcal{F}'$ at key $k = k_0 + k_1$, for any $k_1 \in \mathcal{K}$ and input $x_1 \in \mathcal{Y}$ as the input homomorphism of the PRF family $F \in \mathcal{F}$ allows us to compute: $F'_k(x_0 || x_1) = F_{k_1}(x_0 || x'_1) + F_{k_0}(x_0 || \mathbf{0})$, where $x'_1 \in \mathcal{X}$. The pseudorandomness and security follow from that of our HVL-KIH-PRF family.

8 QPC-UE-UU

In this section, we present the first quantum-safe (Q) post-compromise (PC) secure updatable encryption (UE) scheme with unidirectional updates (UU) as an example application of our KIH-HVL-PRF family. An updatable encryption scheme, **UE**, contains algorithms for a data owner and a host. We begin by recalling the definitions of these algorithms. The owner encrypts the data using a secure encryption algorithm, **UE.enc**, and then outsources the ciphertexts to some host. To this end, the data owner initially runs a key generation algorithm, **UE.setup**, to sample an encryption key. This key evolves with epochs and the data is encrypted with respect to a specific epoch e , starting with $e = 0$. When moving from epoch e to epoch $e + 1$, the owner invokes the update token algorithm **UE.next** to generate the key material k_{e+1} for the new epoch and calculate the corresponding update token Δ_{e+1} . The owner then sends Δ_{e+1} to the host, deletes k_e and Δ_{e+1} immediately, and uses k_{e+1} for encryption from now on. Definition 9 gives a formal description of the procedures and algorithms.

Definition 9 ([38]). An updatable encryption scheme **UE** for message space \mathcal{M} consists of a set of polynomial-time algorithms **UE.setup**, **UE.next**, **UE.enc**, **UE.dec**, and **UE.upd** satisfying the following conditions:

UE.setup: The algorithm **UE:setup** is a probabilistic algorithm run by the owner. On input a security parameter λ , it returns a secret key $k_0 \xleftarrow{\$} \mathbf{UE.setup}(\lambda)$.

UE.next: This probabilistic algorithm is also run by the owner. On input a secret key k_e for epoch e , it outputs the secret key, k_{e+1} , and an update token, Δ_{e+1} , for epoch $e + 1$. That is, $(k_{e+1}, \Delta_{e+1}) \xleftarrow{\$} \mathbf{UE.next}(k_e)$.

UE.enc: This probabilistic algorithm is run by the owner, on input a message $m \in \mathcal{M}$ and key k_e of some epoch e returns a ciphertext $C_e \leftarrow \$ \mathbf{UE.enc}(k_e, m)$.

UE.dec: This deterministic algorithm is run by the owner, on input a ciphertext C_e and key k_e of some epoch e returns $\{m', \perp\} \leftarrow \mathbf{UE.dec}(k_e, C_e)$.

UE.upd: This probabilistic algorithm is run by the host. Given ciphertext C_e from epoch e and the update token Δ_{e+1} , it returns the updated ciphertext $C_{e+1} \leftarrow \mathbf{UE.upd}(\Delta_{e+1}, C_e)$. After receiving Δ_{e+1} , the host first deletes Δ_e . Hence, during some epoch $e + 1$, the update token Δ_{e+1} is available at the host, but the tokens from earlier epochs have been deleted.

8.1 Settings and Notations

Let $i \xleftarrow{\$} \mathcal{X}$ be the identifier for data block d_i . Let $F : \mathcal{K} \times \mathcal{X} \times \mathcal{X} \rightarrow \mathcal{Z}$ and $F' : \mathcal{K} \times \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{Z}$ be the functions defined in Equation 1 and Equation 2, respectively. Let **KeyGen**(λ) be the key generation algorithm

- **QPC-UE-UU.setup**(λ): Generate a random encryption key $k_0 \xleftarrow{\$} \mathbf{F.KeyGen}(\lambda)$, and sample a random nonce $N_0 \xleftarrow{\$} \mathcal{X}$. Set $e \leftarrow 0$, and return the key for epoch $e = 0$ as: $ki_0 = (k_0, N_0)$.
- **QPC-UE-UU.enc**(ki_e, m): Let $i \xleftarrow{\$} \mathcal{X}$ be randomly sampled element that is shared by the host and the owner. Parse $ki_e = (k_e, N_e)$, and return the ciphertext for epoch e as: $C_e = (F'_{k_e}(i, N_e) + m)$. Note that the random nonce ensures that the encryption is not deterministic.
- **QPC-UE-UU.dec**(ki_e, C_e): Parse $ki_e = (k_e, N_e)$. Return $m \leftarrow C_e - F'_{k_e}(i, N_e)$.
- **QPC-UE-UU.next**(ki_e):
 1. Sample a random nonce $N_{e+1} \xleftarrow{\$} \mathcal{X}$. Parse ki_e as (k_e, N_e) .
 2. For epoch $e + 1$, generate a random encryption key $k_{e+1} \xleftarrow{\$} \mathbf{F.KeyGen}(\lambda)$, and return $\Delta_{e+1} = (\Delta_{e+1}^k, \Delta_{e+1}^N)$, where $\Delta_{e+1}^N = N_e \oplus N_{e+1}$ is the nonce update token, and $\Delta_{e+1}^k = k_{e+1} - k_e$ is the encryption key update token. The key for epoch $e + 1$ is $ki_{e+1} = (2k_e - k_{e+1}, N_{e+1})$, where $2k_e - k_{e+1}$ is the encryption key.
- **QPC-UE-UU.upd**(Δ_{e+1}, C_e): Update the ciphertext as: $C_{e+1} = C_e - F'_{\Delta_{e+1}^k}(i, \Delta_{e+1}^N)$
 $= F'_{k_e}(i, N_e) + m - F'_{k_{e+1} - k_e}(i, N_e \oplus N_{e+1})$
 $= F'_{-k_{e+1} + 2k_e}(i, N_e - (N_e \oplus N_{e+1})) + m = F'_{2k_e - k_{e+1}}(i, N_{e+1}) + m.$

Fig. 1: Quantum-safe, post-compromise secure updatable encryption scheme with unidirectional updates, (QPC-UE-UU).

for F , where λ is the security parameter. Figure 1 gives our **QPC-UE-UU** scheme, wherein a random nonce is generated per key rotation, hence ensuring that the encryption remains probabilistic, despite our PRF family being deterministic.

8.2 Proof of Unidirectional Updates

As explained in Section 8.1, the random nonce ensures that the encryption in our scheme is probabilistic. Hence, the security of our QPC-UE-UU scheme follows from the pseudorandomness of our HVL-KIH-PRF family. We move on to proving that the ciphertext updates performed by our scheme are indeed unidirectional. Recall that in schemes with unidirectional updates, an update token Δ_{e+1} can only be used to move ciphertexts from epoch e into epoch $e + 1$, but not vice versa. The notations used in the proof are the same as given in Figure 1.

Lemma 2. *For the HVL-KIH-PRF family, $\mathcal{F}' : \mathcal{K} \times \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{Z}$, as defined in Corollary 4,z with $\mathcal{X} = \{0, 1\}^{|T|}$, $\mathcal{Y} = \{0, 1, \bar{0}\}^{|T|}$ and $\mathcal{Z} = \mathbb{Z}_p^{nd \times nd}$: given ciphertext, C_{e+1} and update token $\Delta_{e+1} = (k_{e+1} - k_e, N_e \oplus N_{e+1})$ for epoch $e + 1$, the following holds for a polynomial adversary \mathcal{A} , randomly sampled $Q \xleftarrow{\$} \mathcal{Z}$ and security parameter λ :*

$$Pr[C_e] = Pr[Q] \pm \epsilon(\lambda),$$

where $\epsilon(\lambda)$ is a negligible function.

Proof. The main idea of the proof is that due to the bi-homomorphic property of our HVL-KIH-PRF family, \mathcal{F}' , the adversary, \mathcal{A} , can only revert back to either the key k_e or the nonce N_e , but not both. In other

words, \mathcal{A} cannot recover $C_e = F'_{k_e}(i, N_e) + m$. We split the proof into two portions, one concentrating on reverting back to k_e as the function key, and the other one focusing on moving back to N_e as the function input. We prove that these two goals are mutually exclusive for our scheme, i.e., both of them cannot be achieved together.

Case 1: Reverting to k_e . Recall from Table 1 that the only well-defined operation for the operand $\bar{0}$ is $\bar{0} = 0 + 0$. We know that the ciphertext update for epoch $e + 1$ is performed as: $C_{e+1} = C_e - F'_{\Delta_{e+1}^k}(i, \Delta_{e+1}^N)$. Since \mathcal{F}' is a PRF family, the only way to revert back to $F'_{k_e} \in \mathcal{F}'$ via C_{e+1} and Δ_{e+1} is by computing:

$$\begin{aligned} C_{e+1} + F'_{\Delta_{e+1}^k}(i, \Delta_{e+1}^N) &= F'_{2k_e - k_{e+1}}(i, N_{e+1}) + m + F'_{k_{e+1} - k_e}(i, N_e \bar{\oplus} N_{e+1}) \\ &= F'_{k_e}(i, N_{e+1} \bar{\oplus} (N_e \bar{\oplus} N_{e+1})). \end{aligned}$$

Due to the key homomorphism exhibited by \mathcal{F}' , no other computations would lead to the target key k_e . We know that $\Delta_{e+1}^N (= N_e \bar{\oplus} N_{e+1}) \in \mathcal{Y}$, and that $N_{e+1}, N_e \in \mathcal{X}$. Therefore, the output of the above computation is not well-defined since it leads to $\Delta_e^N \bar{\oplus} N_{e+1} \notin \mathcal{Y}$ as being the input to $F'_{k_e}(i, \cdot) \in \mathcal{F}'$. Hence, when \mathcal{A} successfully reverts back to the target key k_e , the nonce deviates from N_e (and the domain \mathcal{Y} itself).

Case 2: Reverting to N_e . Given C_{e+1} and Δ_{e+1} , \mathcal{A} can revert back to (i, N_e) as the function input by computing: $C_{e+1} - F'_{\Delta_{e+1}^k}(i, \Delta_{e+1}^N) = F'_{3k_e - 2k_{e+1}}(i, N_e) + m$. By virtue of the almost XOR operation and the operand $\bar{0}$, the only way to revert back to $N_e \in \mathcal{X}$ from $\Delta_{e+1}^N (= N_e \bar{\oplus} N_{e+1}) \in \mathcal{Y}$ is via subtraction. But, as shown above, subtraction leads to $F'_{3k_e - 2k_{e+1}}(i, N_e)$ instead of $F'_{k_e}(i, N_e)$. Hence, the computation that allows \mathcal{A} to successfully revert back to N_e as the function input, also leads the function's key to deviate from k_e .

Considering the aforementioned arguments, it follows from Corollary 4 that $\forall Q \xleftarrow{\$} \mathcal{Z}$, it holds that: $Pr[C_e] = Pr[Q] \pm \epsilon(\lambda)$. \square

9 Open Problem: Novel Searchable Encryption Schemes

Searchable symmetric encryption (SSE) [21] allows one to store data at an untrusted server, and later search the data for records (or documents) matching a given keyword. Multiple works [5, 19, 21, 25, 32, 31, 37, 56, 39] have studied SSE and provided solutions with varying trade-offs between security, efficiency, and the ability to securely update the data after it has been encrypted and uploaded. A search pattern [21] is defined as any information that can be derived or inferred about the keywords being searched from the issued search queries. In a setting with multiple servers hosting unique shares of the data (generated via threshold secret sharing [54]), our HVL-KIH-PRF family may be useful in realizing SSE scheme that hides search patterns. For instance, if there are n servers S_1, S_2, \dots, S_n , then n random keys k_1, k_2, \dots, k_n can be distributed among them in a manner such that any t -out-of- n servers can combine their respective keys to generate $k = \sum_{j=1}^n k_j$.

If the search index is generated via our HVL-KIH-PRF function $F'_k(i, \cdot) \in \mathcal{F}'$, where i is a fixed database identifier, then to search for a keyword x , the data owner can generate a unique, random query x_j for each server S_j ($1 \leq j \leq n$). Similar to the key distribution, the data owner sends the queries to the servers

such that any t of them can compute $x = \bigoplus_{j=1}^n x_j$. On receiving search query x_j , server S_j uses its key k_j to evaluate $F'_{k_j}(i, x_j)$. If at least t servers reply, the data owner can compute $\sum_{j=1}^t F'_{k_j}(i, x_j) = F'_k(i, x)$. Designing a compact search index that does not leak any more information than what is revealed by the PRF evaluation is an interesting open problem, solving which would complete this SSE solution.

10 Conclusion

Key-homomorphic PRFs have found a multitude of interesting applications in cryptography. In this paper, we further expanded the domain of PRFs by introducing bi-homomorphic PRFs, which exhibit full homomorphism over the key and partial homomorphism over the input, which also leads to homomorphically-induced variable input length. We presented a LWE-based construction for such a PRF family, which is inspired by the key-homomorphic PRF construction given by Banerjee and Peikert [7].

We use our novel PRF family to develop the first quantum-safe, post-compromise secure updatable encryption scheme with unidirectional updates. The homomorphically-induced variable input length of our PRF family allowed us to address the open problem of achieving unidirectional updates for updatable encryption. As a special case of our PRF family, we introduced key-homomorphic constrained-PRF with homomorphically-induced variable input length. We leave as an open problem the question of using our PRF family to design search pattern hiding searchable symmetric encryption schemes. With appropriately designed search index, the bi-homomorphism property of our PRF family should allow random search queries, hence leading to private search patterns.

References

1. Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional Encryption for Inner Product Predicates from Learning with Errors. In *ASIACRYPT*, pages 21–40, 2011.
2. Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In *TCC*, pages 474–495, 2009.
3. Joël Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with Rounding, Revisited. In *CRYPTO*, pages 57–74, 2013.
4. Amazon. Protecting data using client-side encryption, 2006.
5. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems. In *CRYPTO*, pages 595–618, 2009.
6. Abhishek Banerjee, Georg Fuchsbauer, Chris Peikert, Krzysztof Pietrzak, and Sophie Stevens. Key-homomorphic constrained pseudorandom functions. In *TCC*, pages 31–60, 2015.
7. Abhishek Banerjee and Chris Peikert. New and Improved Key-Homomorphic Pseudorandom Functions. In *CRYPTO*, pages 353–370, 2014.
8. Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *EUROCRYPT*, pages 719–737, 2012.

9. M. Bellare, R. Canetti, and H. Krawczyk. Pseudorandom functions revisited: the cascade construction and its concrete security. In *FOCS*, pages 514–523, 1996.
10. Mihir Bellare and Phillip Rogaway. On the construction of variable-input-length ciphers. In *FSE*, pages 231–244, 1999.
11. Hayo BaanSauvik Bhattacharya, Scott Fluhrer, Oscar Garcia-Morchon, Thijs Laarhoven, Ronald Rietman, Markku-Juhani O. Saarinen, Ludo Tolhuizen, and Zhenfei Zhang. Round5: Compact and fast post-quantum public-key encryption. In *PQCrypto*, pages 83–102, 2019.
12. Andrej Bogdanov, Siyao Guo, Daniel Masny, Silas Richelson, and Alon Rosen. On the Hardness of Learning with Rounding over Small Modulus. In *TCC*, pages 209–224, 2016.
13. Dan Boneh, Kevin Lewi, Hart Montgomery, and Ananth Raghunathan. Key homomorphic PRFs and their applications. In *CRYPTO*, pages 410–428, 2013.
14. Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *ASIACRYPT*, pages 280–300, 2013.
15. Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional Signatures and Pseudorandom Functions. In *PKC*, pages 501–519, 2014.
16. Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. *STOC*, pages 575–584, 2013.
17. Zvika Brakerski and Vinod Vaikuntanathan. Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages. In *CRYPTO*, pages 505–524, 2011.
18. Zvika Brakerski and Vinod Vaikuntanathan. Constrained Key-Homomorphic PRFs from Standard Lattice Assumptions – or: How to Secretly Embed a Circuit in Your PRF. In *TCC*, pages 1–30, 2015.
19. Yan-Cheng Chang and Michael Mitzenmacher. Privacy preserving keyword searches on remote encrypted data. In *Applied Cryptography and Network Security*, pages 442–455, 2005.
20. Anamaria Costache and Nigel P. Smart. Homomorphic encryption without gaussian noise. *IACR Cryptology ePrint Archive*, 163, 2017.
21. Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In *ACM conference on Computer and communications security*, pages 79–88, 2006.
22. Jan-Pieter D’Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure kem. In *AFRICACRYPT*, pages 282–305, 2018.
23. A. Everspaugh, K.G. Paterson, T. Ristenpart, and S. Scott. Key rotation for authenticated encryptionkey rotation for authenticated encryption. In *CRYPTO*, pages 98–129, 2017.
24. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan Craig Gentry. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.
25. Eu-Jin Goh. Secure indexes. *Cryptology ePrint Archive, Report 2003/216*, 2003.
26. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM (JACM)*, 33:792–807, Oct. 1986.
27. Shafi Goldwasser, Yael Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nikolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In *STOC*, pages 555–564, 2013.
28. Google. Managing data encryption, 2019.
29. N. J. Hopper and M. Blum. Secure human identification protocols. In *ASIACRYPT*, pages 52–66, 2001.
30. A. Juels and S. A. Weis. Authenticating pervasive devices with human protocols. In *CRYPTO*, pages 293–308, 2005.
31. Seny Kamara and Charalampos Papamanthou. Parallel and Dynamic Searchable Symmetric Encryption. In *Financial Cryptography and Data Security*, pages 258–274, 2013.
32. Seny Kamara, Charalampos Papamanthou, and Tom Roeder. Dynamic searchable symmetric encryption. In *CCS*, pages 965–976, 2012.
33. J. Katz, J.S. Shin, and A. Smith. Parallel and concurrent security of the HB and hb+ protocols. *Journal of Cryptology*, 2010.

34. Jonathan Katz and Vinod Vaikuntanathan. Smooth projective hashing and password-based authenticated key exchange from lattices. In *ASIACRYPT*, pages 636–652, 2009.
35. Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In *CCS*, pages 669–684, 2013.
36. E. Kiltz, K. Pietrzak, D. Cash, A. Jain, and D. Venturi. Efficient authentication from hard learning problems. In *EUROCRYPT*, pages 7–26, 2011.
37. Kaoru Kurosawa and Yasuhiro Ohtaki. UC-Secure searchable symmetric encryption. In *Financial Cryptography and Data Security*, pages 285–298, 2012.
38. Anja Lehmann and Björn Tackmann. Updatable encryption with post-compromise security. In *EUROCRYPT*, pages 685–716, 2018.
39. Peter Van Liesdonk, Saeed Sedghi, Jeroen Doumen, Pieter Hartel, and Willem Jonker. Computationally Efficient Searchable Symmetric Encryption. In *Workshop on Secure Data Management*, pages 87–100, 2010.
40. Vadim Lyubashevsky. Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In *ASIACRYPT*, pages 598–616, 2009.
41. Vadim Lyubashevsky. Lattice Signatures without Trapdoors. In *EUROCRYPT*, pages 738–755, 2012.
42. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *EUROCRYPT*, 2010.
43. Daniele Micciancio and Chris Peikert. Hardness of sis and lwe with small parameters. In *CRYPTO*, pages 21–39, 2013.
44. Moni Naor, Benny Pinkas, and Omer Reingold. Distributed Pseudo-random Functions and KDCs. In *EUROCRYPT*, pages 327–346, 1999.
45. Moni Naor and Omer Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. *Journal of Computer and System Sciences*, pages 336–375, 1999.
46. Jhordany Rodriguez Parra, Terence Chan, and Siu-Wai Ho. A noiseless key-homomorphic prf: Application on distributed storage systems. In *Australasian Conference on Information Security and Privacy*, pages 505–513, 2016.
47. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. *STOC*, pages 333–342, 2009.
48. Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC*, pages 145–166, 2006.
49. Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A Framework for Efficient and Composable Oblivious Transfer. In *CRYPTO*, pages 554–571, 2008.
50. K. Pietrzak. Subspace LWE. In *TCC*, pages 548–563, 2012.
51. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93, 2005.
52. Markus Rückert. Lattice-based blind signatures. In *ASIACRYPT*, pages 413–430, 2010.
53. Vipin Singh Sehrawat and Yvo Desmedt. Bi-Homomorphic Lattice-Based PRFs and Unidirectional Updatable Encryption. In *CANS*, volume 11829, pages 3–23. LNCS, Springer, 2019.
54. Adi Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, Nov. 1979.
55. P. W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *FOCS*, pages 124–134, 1994.
56. Dawn Xiaodong Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In *IEEE Symposium on Security and Privacy*, pages 44–55, 2000.